REMARKS

Claims 1-14 and 17-32 are now pending in this application. In the May 27, 2004 Office Action, claims 1-14 and 17-32 were rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,269,254 to Mathis (hereinafter "*Mathis*").

For the reasons set forth below, the applicants respectfully request reconsideration and immediate allowance of this application. Prior to discussing the reasons why the applicants believe that the claims currently pending in this application are allowable, a brief description of the present invention and the cited reference is presented.

Summary of the Invention

The present invention provides a Radio Interface Layer (RIL) between the radio on a cell phone and the software of the cell phone. The RIL is an Application Program Interface (API) set that provides a level of abstraction between the radio and software of a cell phone. According to one embodiment, the RIL comprises a proxy layer and a driver layer. The RIL proxy layer is located just below application modules for communication with the applications. The RIL driver layer is located between the RIL proxy layer and the radio hardware to communicate with the proxy layer and radio hardware. The RIL allows communication between the application modules and the radio hardware by exposing a set of functions. The application modules use these functions to communicate with the proxy layer through a first interface. The proxy layer and driver layer communicate through a second interface. The driver layer communicates with the radio hardware through a third interface.

The proxy layer is hardware-independent while the driver layer is hardware-specific. For this reason, the RIL provides a platform for third party software developers. Software applications may be easily written to work with the RIL because the applications use well-known top-level APIs which are sent to the RIL. The RIL will then perform appropriate processing of these top-level APIs and, if necessary, send the appropriate command to the radio to perform a specific action. Similarly, radio manufacturers no longer have to worry about receiving and keeping track of calls from multiple client applications because all of these functions are handled by the RIL. Without the RIL, each component (i.e., TAPI, SIM manager, SMS manager, etc.) of

the phone would have to understand how to communicate to the radio directly. Because it would be difficult for hardware manufacturers to implement a TAPI driver, a SMS driver, a SIM driver, etc., the RIL was created to sit between the radio and the TAPI driver, the SMS driver, the SIM driver, etc.

Summary of the Cited Reference

Mathis describes supporting a dual-mode call on a GSM radio telephone using a Java telephony API (JTAPI). *Mathis* teaches using a JTAPI to interface Java application programs to service providers or to existing telephony APIs (TAPIs). These service providers and existing TAPIs communicate directly with the radio hardware. Where GSM functions cannot be easily accessed using the existing JTAPI syntax and method signature, *Mathis* defines new methods or methods with different signatures. *Mathis* relies on applications written using JTAPI to achieve the desired dual mode call functionality. As will be discussed below, *Mathis* teaches specific use of JTAPI to allow applications to communicate with service providers and existing TAPIs and does not teach proxy layer and driver layer interfaces between the service provider and TAPIs to communicate between the service provider and TAPIs and the underlying radio.

Claim Rejections - 35 U.S.C. §102 - *Mathis*

The Examiner rejected claims 1-14 and 17-32 under 35 U.S.C. § 102(e) as being anticipated by *Mathis*. The applicants respectfully submit that *Mathis* fails to teach, suggest, or describe each element of the embodiments of the present invention recited in claims 1-14 and 17-32.

Independent Claim 1

Mathis does not teach, suggest, or describe each recitation of independent claim 1. The applicants have amended independent claim 1 to clarify the communication flow and command transformation through the abstraction layer of the recited embodiment. Specifically, *Mathis* does not teach, suggest, or describe, "wherein the abstraction layer comprises a proxy layer and a driver layer, wherein when the proxy layer receives a call at a first interface to one of the set of

APIs, the proxy layer transforms the API call to a command understood by the driver layer and sends the command to the driver layer at a second interface, and wherein the driver layer receives the command at the second interface and determines at least one standard telephony radio command corresponding to the called API and sends the telephony radio command to the telephony radio at a third interface."

The applicants respectfully submit that *Mathis* does not teach, suggest, or describe an abstraction layer that comprises "a proxy layer and a driver layer" as set forth in claim 1. The May 27, 2004 Office Action suggests that the architecture of the JTAPI "includes a proxy 'server' and an underlying driver." See page 8, item 40 of Office Action. Referring to the cited passage, the Office Action equates the telephony server implementations to the proxy layer of the present invention. The applicants respectfully disagree since the cited passage states, "telephony server implementations choose which extension packages they implement, based upon the capabilities of the underlying hardware." *Mathis*, col. 12, lines 32-34. One advantage of the proxy layer recited by the present application is that the proxy layer is hardware independent. The proxy layer of claim 1 "transforms the API call to a command understood by the driver layer" so that the proxy layer does not need to have any knowledge or understanding as to the underlying hardware. The driver layer is used to "receive[s] the command at the second interface and determine[s] at least one standard telephony radio command corresponding to the called API and send[s] the telephony radio command to the telephony radio at a third interface" as recited by claim 1.

Moreover, the applicants submit that *Mathis* does not teach, suggest, or describe a driver, specifically a driver layer that "receives the command at the second interface and determines at least one standard telephony radio command corresponding to the called API and sends the telephony radio command to the telephony radio at a third interface" as recited by claim 1. The applicants disagree with the assertion made in the May 27, 2004 Office Action that *Mathis* "inherently" teaches a driver. Although drivers may be used in various situations to provide for communication between software and a piece of hardware, drivers are not required. Any particular software application may be written to communicate with a piece of hardware,

provided that the software uses the correct commands and controls applicable to that particular piece of hardware.

The applicants would like to refer the Examiner to FIG. 2 of the present application. This figure illustrates how the proxy layer and driver layer provide a level of abstraction between radio hardware and application modules. The RIL proxy layer receives a call at a first interface to one of the set of APIs from any of the application modules shown. The proxy layer transforms the API call to a command understood by the driver layer and sends the command to the driver layer at a second interface. The driver layer receives the command at the second interface and determines at least one standard telephony radio command corresponding to the called API and sends the telephony radio command to the telephony radio at a third interface. Now referring to FIG. 8 of *Mathis*, the applicants submit that *Mathis* does not teach a proxy layer and a driver layer directly interfaced with each other and the radio hardware. For at least these reasons, independent claim 1 is allowable over *Mathis*.

Independent Claim 7

Mathis does not teach, suggest, or describe each recitation of independent claim 7. The applicants have amended independent claim 7 to clarify the communication flow and command transformation through the abstraction layer of the recited embodiment. In particular, *Mathis* does not teach, suggest, or describe "a proxy layer for communicating with the application program at a first interface and a driver layer at a second interface, wherein the proxy layer provides an API on the first interface for receiving application program calls to perform a particular function and wherein the proxy layer transforms the API calls to an input/output control (IOCTL) code and sends the IOCTL code to the driver layer at the second interface; and wherein the driver layer communicates with the proxy layer at the second interface and the radio at a third interface, the driver layer receiving an IOCTL code at the second interface and transforming the IOCTL code into a command understood by the radio to perform the particular function and sending the radio command at the third interface" as recited by amended claim 7.

For the same reasons discussed above with respect to independent claim 1, the applicants submit that *Mathis* does not teach a proxy layer and driver layer that are directly interfaced with

each other and the radio hardware. The May 27, 2004 Office Action further suggests the "Java RUN-TIME machine" described by *Mathis* and shown in FIG. 8 is equivalent to the proxy layer recited by claim 7. The applicants respectfully submit that the Java RUN-TIME machine of FIG. 8 simply enables the Java code of the application to run. The Java RUN-TIME machine does not receive a call at a first interface to one of the set of APIs, transform the API call to a command understood by the driver layer, and send the command to the driver layer at a second interface as recited by amended claim 1. As illustrated by FIG. 8 of *Mathis*, the Java RUN-TIME machine is located between the JTAPI and TAPI layers. This contrasts the RIL recited by the present invention that is located between the application modules and the hardware.

Further, the applicants submit that *Mathis* does not teach a driver layer that receives the IOCTL code from the proxy layer and transforms it into a command understood by the radio to perform the particular function. The IOCTL functions are not performed by the CSPMI as suggested by the May 27, 2004 Office Action. As shown in FIG. 3 of *Mathis*, the CSPMI is used to communicate with the GSM transceiver. In the aspect recited by claim 7, the IOCTL code is used to communicate with the driver, which then transforms it into a command understood by the radio prior to transmitting it to the radio. For at least these reasons, independent claim 7 is allowable over *Mathis*.

Independent Claim 9

Mathis does not teach, suggest, or describe each recitation of independent claim 9. The applicants have amended independent claim 9 to clarify the communication flow and command transformation through the abstraction layer of the recited embodiment. In particular, *Mathis* does not teach, suggest, or describe, "causing the application to call a radio interface layer (RIL) API in the proxy layer at a first interface, wherein the RIL API is associated with an action to be performed by the radio; causing the proxy layer to translate the RIL API into IOCTL codes; sending the IOCTL codes to the driver layer at a second interface; translating the IOCTL codes to a command corresponding to the action, wherein the command will be understood by the radio; and sending the command to the radio at a third interface" as recited by claim 9. For at

least the reasons discussed above with respect to independent claims 1 and 7, independent claim 9 is allowable over *Mathis*.

Independent Claim 17

Mathis does not teach, suggest, or describe each recitation of independent claim 17. The applicants have amended independent claim 17 to clarify the communication flow and command transformation through the abstraction layer of the recited embodiment. In particular, *Mathis* does not teach, suggest, or describe "(b) sending the RIL API call to a proxy at a first interface; (c) at the proxy, converting the RIL API call to a command understood by a radio driver; (d) transmitting the radio driver command from the proxy to the radio driver at a second interface; (e) transmitting a radio command from the radio driver to the radio at a third interface" as recited by claim 17. For at least the reasons discussed above with respect to independent claim 1, independent claim 17 is allowable over *Mathis*.

Dependent Claims 2-6, 8, 10-14, and 18-32

Because *Mathis* does not teach, suggest, or describe every element of claims 2-6, 8, 10-14, and 18-32 and because claims 2-6, 8, 10-14, and 18-32 depend from allowable independent claims 1, 7, 9, and 17, dependent claims 2-6, 8, 10-14, and 18-32 are allowable over *Mathis*.

CONCLUSION

In view of the foregoing amendment and remarks, the applicants respectfully submit that the present application is in condition for allowance. Reconsideration and reexamination of the application and allowance of the claims at an early date is solicited. If the Examiner has any questions or comments concerning this matter, the Examiner is invited to contact the applicants' undersigned attorney at the number below.

Respectfully submitted,

MERCHANT & GOULD

Leonard J. Hope

Date: July 22, 2004

Leonard J. Hope
Reg. No.: 44,774

Merchant & Gould, LLC
P.O. Box 2903
Minneapolis, Minnesota 55402-0903
Telephone: 404.954.5100